Implementing The Simplex Method: The Initial Basis

Robert E. Bixby [1]
Department of Mathematical Sciences
Rice University

September 1991
(revised)
TR90-32

[1]The preparation of this manuscript was supported in part by NSF grant CCR-8815914 and AFOSR grant AFOSR-90-0273, both to Rice University.

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **SEP 1991**	2. REPORT TYPE	3. DATES COVERED **00-00-1991 to 00-00-1991**
4. TITLE AND SUBTITLE **Implementing The Simplex Method: The Initial Basis**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Computational and Applied Mathematics Department ,Rice University,6100 Main Street MS 134,Houston,TX,77005-1892**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**29**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Abstract

This paper contains the first two parts of a planned series of papers on the CPLEX[2] implementation of the simplex method. Part I is introductory. It gives an elementary description of the bounded-variable simplex method as well as a rather detailed discussion of some of the numerical characteristics of the netlib test problems. These problems form the basis for the computational tests in the subsequent parts. Part II contains the main results of this paper, a description of the method used by CPLEX for constructing an initial basis.

[2]CPLEX is a trademark of CPLEX Optimization, Inc.

Part I: Introduction

1 Introduction

Ten years ago, linear programming was considered a computationally mature subject. Dantzig's simplex method [3] was thought to be well understood, and there were several state-of-the-art implementations of that method and its extensions; moreover, it is probable that the majority of the mathematical-programming community did not expect substantial improvements. That expectation was incorrect. The last ten years have seen dramatic changes in computing machinery. These changes have allowed a wider variety of strategies to be implemented and much larger problems to be studied in detail. Developments have also been driven by demands from other areas of optimization, particularly integer programming, where the availability of a good linear optimizer is essential. In addition, motivated by work of Karmarkar [6], a new class of methods, interior-point methods, has emerged as an alternative to the simplex method. Although interior-point methods are not treated in this paper, the recent computational results have been every bit as impressive as those for the simplex method ([1], [9]).

This paper is the first in a series of papers describing different aspects of the CPLEX implementation of the simplex method, the optimization routines for which were written by this author. Two other excellent, recent implementations of the simplex method are the OSL (Optimization System Library) implementation [5], written by John Forrest, and MINOS 5.3, written by Michael Saunders [11], the latter being a significant improvement over earlier versions of MINOS.

The main purpose of the present paper, and the subject of Part II, is to describe the procedure used by CPLEX for constructing a good initial basis. Part I provides a foundation for that discussion. It includes the definition of a basis and a statement of the bounded-variable simplex method, followed by a discussion of the numerical properties of the netlib test problems. Some readers more familiar with linear programming computation may wish to skip directly to Part II.

In subsequent papers in this series, other CPLEX features will be discussed, including the use of Philip Wolfe's composite simplex method in phase I, an inexpensive partial-pricing procedure, a simple bound-perturbation for dealing with stalling, vectorization and other topics.

2 The Definition of a Basis

A *linear programming problem* (LP) is an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{s.t.} & Ax = b \\ & l \le x \le u \end{array} \tag{1}$$

where c, A, b, l and u are given matrices with dimensions $n \times 1$, $m \times n$, $m \times 1$, $n \times 1$ and $n \times 1$, respectively, and x is an $n \times 1$ vector of variables. Assume that A has full row rank. When this condition is not met, appropriate unit columns can be added.

The linear functional $c^T x$ is called the *objective function*, A the *constraint matrix*, b the *right-hand side*, l the vector of *lower bounds* and u the vector of *upper bounds.* The "values" $-\infty$ for lower bounds and $+\infty$ for upper bounds are allowed. Most x_j have $l_j = 0$ and $u_j = +\infty$, and are called *nonnegative.* If $l_j = -\infty$ and $u_j = +\infty$, x_j is *free*, and if $l_j = u_j$, x_j is *fixed.*

If S a subset of columns of A, A_S denotes "the" $m \times |S|$ submatrix containing the columns in S. If S is an ordered set, the columns of A_S are taken to appear in this order. If d is a vector and S is a subset of row indices, then d_S is the corresponding subvector. Again, if S is ordered, the rows of d_S are taken ordered.

Definition. A *basis* is a triple (B, N_l, N_u) with the following two properties:

(B1) $B = (B_1, \ldots, B_m) \subseteq \{1, \ldots, n\}$ is an ordered subset of column indices such that $\boldsymbol{B} = A_B$ is nonsingular. B is called the *basis header* and $\boldsymbol{B}$ the *basis matrix.* The variables x_j $(j \in B)$ are called *basic variables.* The remaining variables are *nonbasic.* $N = \{j : j \notin B\}$ denotes the set of indices of nonbasic variables.

(B2) $N_l \cap N_u = \emptyset$, $N_l \cup N_u = \{j \notin B : x_j \text{ neither fixed nor free}\}$, $l_j > -\infty$ for $j \in N_l$ and $u_j < +\infty$ for $j \in N_u$.

Denote $N_{fr} = \{j \in N : x_j \text{ free}\}$ and $N_{fx} = \{j \in N : x_j \text{ fixed}\}$. $\square$

The triple (B, N_l, N_u) is sometimes referred to as "the basis B." The set B is assumed ordered to simplify the statement of the algorithm. There is no corresponding need to order N_l or N_u.

Corresponding to each basis there is a *basic solution* X given by

$$\begin{array}{lcl} X_{N_l} & = & l_{N_l}, \\ X_{N_u} & = & u_{N_u}, \\ X_{N_{fx}} & = & l_{N_{fx}} = u_{N_{fx}}, \\ X_{N_{fr}} & = & 0, \text{ and} \\ X_B & = & \boldsymbol{B}^{-1}(b - A_N X_N). \end{array} \tag{2}$$

The basis B is called *feasible* if $l_B \le X_B \le u_B$.

3 The Bounded-Variable Simplex Method

Algorithm 3.1. A Single Iteration of a Simplex Algorithm for (1)

Input: A feasible basis B and the corresponding values X_B of the basic variables.

Step S1: Solve $\pi^T \boldsymbol{B} = c_B^T$ for π.

Step S2: (Pricing) Compute all or part of $d_N = c_N - A_N^T \pi$. If $d_j \geq 0$ for all $j \in N_l$, $d_j \leq 0$ for all $j \in N_u$ and $d_j = 0$ for all $j \in N_{fr}$, stop—B is optimal; otherwise, select an *entering variable* x_{j_e}, $j_e \in N$, such that d_{j_e} violates these conditions.

Step S3: Solve $\boldsymbol{B}y = A_{j_e}$.

Step S4: (Ratio Test) If $d_{j_e} < 0$, let

$$\Theta_i = \begin{cases} +\infty & \text{if } \ y_i = 0, \\ (X_{B_i} - l_{B_i})/y_i & \text{if } \ y_i > 0, \text{ and} \\ (X_{B_i} - u_{B_i})/y_i & \text{if } \ y_i < 0, \end{cases}$$

and if $d_{j_e} > 0$, let

$$\Theta_i = \begin{cases} +\infty & \text{if } \ y_i = 0, \\ (u_{B_i} - X_{B_i})/y_i & \text{if } \ y_i > 0, \text{ and} \\ (l_{B_i} - X_{B_i})/y_i & \text{if } \ y_i < 0 \end{cases}$$

for $i = 1, \ldots, m$. Let

$$\Theta = \min\{\min_i \Theta_i, u_{j_e} - l_{j_e}\}.$$

If $\Theta = +\infty$, stop—(1) is unbounded.

Step S5: (Update) If $d_{j_e} < 0$, set $X_B \leftarrow X_B - \Theta y$; otherwise, set $X_B \leftarrow X_B + \Theta y$.

(S5.1) For $\Theta = u_{j_e} - l_{j_e}$: If $j_e \in N_u$, set $N_l \leftarrow N_l \cup \{j_e\}$ and $N_u \leftarrow N_u \backslash \{j_e\}$; otherwise, set $N_u \leftarrow N_u \cup \{j_e\}$ and $N_l \leftarrow N_l \backslash \{j_e\}$.

(S5.2) For $\Theta < u_{j_e} - l_{j_e}$: Let $i_l \in B$ be such that $\Theta_{i_l} = \Theta$, and let $j_l = B_{i_l}$ (x_{j_l} is the *leaving variable*). Set $B_{i_l} \leftarrow j_e$, and set

$$X_{B_{i_l}} \leftarrow \begin{cases} l_{j_e} + \Theta & \text{if } \ j_e \in N_l, \\ u_{j_e} - \Theta & \text{if } \ j_e \in N_u, \\ \Theta & \text{if } \ j_e \in N_{fr} \text{ and } d_{j_e} < 0, \text{ and} \\ -\Theta & \text{if } \ j_e \in N_{fr} \text{ and } d_{j_e} > 0. \end{cases}$$

If x_{j_l} is fixed, set $N_{fx} \leftarrow N_{fx} \cup \{j_l\}$; otherwise, if $d_{j_e} y_{i_l} < 0$, set $N_l \leftarrow N_l \cup \{j_l\}$, and if $d_{j_e} y_{i_l} > 0$, set $N_u \leftarrow N_u \cup \{j_l\}$. Remove j_e from $N_{fr} \cup N_l \cup N_u$. $\square$

Problem (1) is *unbounded* if for an arbitrary integer M, there is an x feasible for (1) such that $c^T x < M$. A basis B is *optimal* for (1) if it is feasible and $c^T X \leq c^T x$ for all feasible x, where X is the basic solution corresponding to B.

The following theorem gives all the properties of Algorithm 3.1 that are necessary to deduce it's validity, namely: that it can be applied iteratively and that its termination criteria are correct. In addition, it is noted that the sequence of objective-function values produced is monotone. On the other hand, finite termination is not proved, nor is that conclusion valid for the given statement of the algorithm [2].

Theorem. *If Algorithm 3.1 terminates in (S2), then B is an optimal basis for (1). If termination occurs in (S4), then (1) is unbounded. Otherwise, the B and X_B produced in (S5) satisfy the input conditions, and the corresponding objective-function value is no bigger than for the initial X.*

Proof. (Optimality) By assumption, B is feasible. Suppose termination occurs in (S2). Let $d = c - A^T\pi$, where π is as computed in (S1). Now for any feasible x, $d^T x = c^T x - \pi^T A x = c^T x - \pi^T b$. Thus, minimizing $d^T x$ is equivalent to minimizing $c^T x$. But for x feasible, one has

$$\begin{aligned} d^T x &= d_B^T x_B + d_{N_l}^T x_{N_l} + d_{N_u}^T x_{N_u} + d_{N_{fr}}^T x_{N_{fr}} + d_{N_{fx}}^T x_{N_{fx}} \\ &= d_B^T X_B + d_{N_l}^T x_{N_l} + d_{N_u}^T x_{N_u} + d_{N_{fr}}^T X_{N_{fr}} + d_{N_{fx}}^T X_{N_{fx}} \\ &\geq d_B^T X_B + d_{N_l}^T l_{N_l} + d_{N_u}^T u_{N_u} + d_{N_{fr}}^T X_{N_{fr}} + d_{N_{fx}}^T X_{N_{fx}} \\ &= d^T X. \end{aligned}$$

The second equality follows because $d_B = 0$, $d_{N_{fr}} = 0$ and $X_{N_{fx}} = x_{N_{fx}}$. The inequality follows because $d_{N_l} \geq 0$, $x_{N_l} \geq l_{N_l}$, $d_{N_u} \leq 0$ and $x_{N_u} \leq u_{N_u}$. Hence, B is optimal.

(Unboundedness) Suppose that $d_{j_e} < 0$, and for $\alpha \geq 0$ define

$$\begin{aligned} X_{j_e}^\alpha &= X_{j_e} + \alpha, \\ X_B^\alpha &= X_B - \alpha y, \text{ and} \\ X_j^\alpha &= X_j \text{ for } j \notin B \cup \{j_e\}, \end{aligned}$$

where $\boldsymbol{B} y = A_{j_e}$, and X is given by (2). It is straightforward to see that $AX^\alpha = b$. Suppose that $\alpha \leq \Theta$. Then clearly $l_{j_e} \leq X_{j_e}^\alpha \leq u_{j_e}$, and so $l_N \leq X_N^\alpha \leq u_N$; moreover, $0 \leq \alpha \leq \min_i \Theta_i$ implies $l_B \leq X_B^\alpha \leq u_B$. Hence, X^α is feasible, and

$$\begin{aligned} c^T X^\alpha &= c^T X + \alpha c_{j_e} - \alpha c_B^T y \\ &= c^T X + \alpha c_{j_e} - \alpha \pi^T A_{j_e} \\ &= c^T X + \alpha d_{j_e} \end{aligned}$$

It follow that if $\Theta = +\infty$, then (1) is unbounded. A similar argument applies when $d_{j_e} > 0$, taking $X_{j_e}^\alpha = X_{j_e} - \alpha$ and $X_B^\alpha = X_B + \alpha y$.

(Update) Finally, it must be shown that if neither optimality nor unboundedness occurs, then the new B is a feasible basis with corresponding values given by the new X_B. Note that the new X_B is nothing but X_B^α, where $\alpha = \Theta$ and X^α is as defined in the unboundedness proof. Hence, the new X_B does give the vector of values of the basic variables, if the new B is a basis; moreover, $c^T X^\Theta \le c^T X$ by the choices of d_{j_e} and Θ.

If (S5.1) occurs, the new B is the old B, and so obviously a basis. For the remaining case, it must be shown that the new $\boldsymbol{B}$ is nonsingular. Denoting the new $\boldsymbol{B}$ by $\bar{\boldsymbol{B}}$, it follows that

$$\bar{\boldsymbol{B}} = \boldsymbol{B}E$$

where

$$E = I + (y - e_{i_l})e_{i_l}^T,$$

I is an $m \times m$ identity, and e_{i_l} is a unit vector with a 1 in row i_l. But $\boldsymbol{B}$ is nonsingular, so $\boldsymbol{B}Ev = 0$ implies

$$v + v_{i_l}(y - e_{i_l}) = Ev = 0,$$

and hence that $v = 0$, since $y_{i_l} \neq 0$ by the definition of Θ. It follows that $\bar{\boldsymbol{B}}$ is nonsingular. $\square$

4 The Netlib Test Problems

The computational tests reported here make use of the *netlib* problems.[3] To the knowledge of this author, the netlib set does not contain any randomly generated problems. All are at least real in the sense that they were humanly generated and represent "real attempts" at modeling.

Some general comments on netlib are probably in order. First, many of the problems in the set are there because they have caused difficulties for the simplex method. Such problems are useful for the purpose of developing robust code, but may not be the best for testing speed and choosing default parameter settings. Second, there are several *classes* of problems on the list. Two examples are pilot (pilot4, perold, pilotwe, pilotnov, pilotja, pilots and pilot87) and fit (fit1p, fit1d, fit2p and fit2d). These classes do help illustrate how performance changes as problem size increases; however, in practice it does not make much sense to run all the problems in such a class without first doing some testing, based on smaller instances in the class. For example, the 'p' and 'd' versions from the fit class represent equivalent, dual formulations of the same problem. After having solved fit1p and fit1d, it is obvious that the 'd' formulation is much easier for the simplex method. From a purely practical viewpoint, there is then little reason to consider solving the fit2p. Similarly, it is well known that the pilot problems are "poorly scaled." The solution

[3] Copies of these problems, assembled by D. M. Gay, can be obtained (at this writing) by "anonymous ftp" to *research.att.com* (userid: *anonymous*, password: <*blank*>). The actual data is contained in the directory *dist/lpdata*.

times on all these problems can be easily improved by using a more aggressive scaling approach.[4]

Three tables of netlib data are given at the end of this section (Tables I, II and III). Table I contains problem data and Tables II and III numerical information on optimal bases, as computed by CPLEX. Blanks indicate zeros. Problems are ordered by the number of constraint nonzeros. For "pictures" of the individual netlib problems see [7] or [8]. (The technical report [7] contains significantly more data than the published version.)

In Table I, counts are given for the various row (constraint) and column (variable) types. *Boxed* variables are variables x_j with bounds of the form $l_j \leq x_j \leq u_j$ where both l_j and u_j are finite. *Fixed* variables are just that, fixed ($l_j = u_j$). As one might expect, nonnegative variables dominate.

In addition to the standard row types, ranged and free rows are listed. *Ranged* rows correspond to constraints of the form

$$\underline{b}_i \leq a_i^T x \leq \overline{b}_i$$

and are modeled by introducing a variable, say y_i, with appropriate bounds, and using this variable to replace the above pair of constraints with a single equality constraint:

$$\begin{gathered} a_i^T x - y = 0 \\ \underline{b}_i \leq y_i \leq \overline{b}_i \end{gathered}$$

Free or unconstrained rows are typically added by users as accounting devices or as a way of encoding alternate objective functions. They are ignored during optimization.

One of the first considerations encountered in implementing virtually any numerical algorithm is the setting of tolerances, frequently replacements for some of the 0's that occur in tests in the mathematical description. (A careful discussion of tolerance settings is beyond the scope of the present paper.) Two such tolerances for the simplex method are the *feasibility* and *optimality tolerances*, used, respectively, to determine whether basic variables satisfy their bounds and whether the vector d of reduced costs satisfies the optimality conditions given in Step S2. The default CPLEX values for the feasibility and optimality tolerances are 1.0E-6 (0.000001); these tolerances can be independently reset to values as low as 1.0E-9.

A third important tolerance is the "pivot threshold" used in computing an LU-factorization of $\boldsymbol{B}$. The factorization algorithm used in CPLEX is based on the method of Markowitz [10] and is implemented as described in [12]. In this scheme, the elimination process is carried out using a relaxation of partial pivoting [4] called *threshold pivoting*. The threshold is a number $0 < \alpha \leq 1$. A pivot element is considered acceptable if it is at least α times the maximum absolute value of a nonzero in the column or row being eliminated. CPLEX takes as the default value $\alpha = 0.01$;

[4]The default scaling used by CPLEX is very simple. First, every row is divided by the maximum absolute value in that row. When that step is complete, every column is divided by the maximum absolute value in that column.

this tolerance can be reset to any value between 0.0001 and 0.99999. (Setting $\alpha = 1.0$ is not allowed for numerical reasons.)

To obtain objective values for Tables II and III that were as accurate as possible, all problems were first solved using default settings and then reoptimized from that point with optimality and feasibility tolerances set to 1.0E-9 and threshold $\alpha = 0.99999$. Computations were carried out on a SPARCstation 2 using IEEE floating-point arithmetic (implying between 15 and 17 digits of accuracy). Data are given for both the solution obtained with defaults and the solution after reoptimization with the indicated "tighter" settings. Data are also presented for both scaled and unscaled versions of the optimal bases. However, it should be noted that no problems were actually *solved* in unscaled form. CPLEX does not permit that.

The condition numbers [4] reported in Table II were computed in the L_∞-norm using an LU-factorization of the reoptimized optimal basis. The relative objective error is given by the formula $|(z_d - z_r)/z_r|$, where z_d is the objective value obtained running defaults and z_r is the value after reoptimization. The "Default Settings Objective Value" column in Table II is identical to the first 11 digits of the "Objective Value" column in Table I.

Residuals reported are maximum absolute deviations: $||AX - b||_\infty$ and $||c_B^T - \pi^T \boldsymbol{B}||_\infty$, where X is as defined in (2), and the system $\pi^T \boldsymbol{B} = c_B^T$ is taken from Step S1 of Algorithm 2.2. The entries in the bound and reduced-cost columns were computed as follows:

$$\begin{aligned}
\text{bound residual} &= \max\{\max\{X_j - u_j, l_j - X_j, 0\} : j \in B\}, \text{ and} \\
\text{reduced-cost residual} &= \max\{\Delta_l, \Delta_u, \Delta_f\},
\end{aligned}$$

where

$$\begin{aligned}
\Delta_l &= \max\{\max\{c_j - \pi^T A_j, 0\} : j \in N_l\}, \\
\Delta_u &= \max\{\max\{\pi^T A_j - c_j, 0\} : j \in N_u\}, \text{ and} \\
\Delta_f &= \max\{|c_j - \pi^T A_j| : j \in N_{fr}\}.
\end{aligned}$$

and the expressions for Δ_l and Δ_u use the fact that all netlib problems have been formulated as minimization problems. As previously indicated, the default tolerances for the bound and reduced-cost residuals are 1.0E-6. Residuals must in each case be no bigger than the corresponding tolerance for the solution to be designated as optimal.

TABLE I

Netlib Problem Data

PROBLEM	Optimal Value	Cons.	Vars.	Nonzeros Cons.	Nonzeros Obj.	Nonzeros RHS	Constraint Types Less	Constraint Types Greater	Constraint Types Equal	Constraint Types Range	Variable Types Nonneg.	Variable Types Box	Variable Types Free	Variable Types Fix	Variable Types Other
1 afiro	-4.6475314286E+02	27	32	83	5	7	19		8		32				
2 sc50b	-7.0000000000E+01	50	48	118	1	5	30		20		48				
3 sc50a	-6.4575077059E+01	50	48	130	1	10	30		20		48				
4 sc105	-5.2202061212E+01	105	103	280	1	20	60		45		103				
5 kb2	-1.7499001299E+03	43	41	286	5	0	12	15	16		32	9			
6 adlittle	2.2549496316E+05	56	97	383	82	37	40	1	15		97				
7 scagr7	-2.3313898243E+06	129	140	420	133	53	38	7	84		140				
8 stocfor1	-4.1131976219E+04	117	111	447	27	8	48	6	63		111				
9 blend	-3.0812149846E+01	74	83	491	30	8	31		43		83				
10 sc205	-5.2202061212E+01	205	203	551	1	38	114		91		203				
11 recipe	-2.6661600000E+02	91	180	663	89	0	6	18	67		85	69		26	
12 share2b	-4.1573224074E+02	96	79	694	36	24	83		13		79				
13 vtpbase	1.2983146246E+05	198	203	908	6	59	133	10	55		87	65	1	18	32
14 lotfi	-2.5264706062E+01	153	308	1078	8	49	42	16	95		308				
15 share1b	-7.6589318579E+04	117	225	1151	31	103	28		89		225				
16 boeing2	-3.1501872802E+02	166	143	1196	143	39	1	142	23	19	89	54			
17 scorpion	1.8781248227E+03	388	358	1426	282	76	48	60	280		358				
18 bore3d	1.3730803942E+03	233	315	1429	96	0	19		214		302	11		1	1
19 scagr25	-1.4753433061E+07	471	500	1554	475	179	146	25	300		500				
20 sctap1	1.4122500000E+03	300	480	1692	360	154		180	120		480				
21 capri	2.6900129138E+03	271	353	1767	19	130	75	54	142		192	131	14	16	
22 brandy	1.5185098965E+03	220	249	2148	2	54	54		166		249				
23 israel	-8.9664482186E+05	174	142	2269	89	171	174				142				
24 finnis	1.7279106560E+05	497	614	2310	404	116	302	148	47		492	36		45	41
25 gfrdpnc	6.9022359995E+06	616	1092	2377	1090	68	68		548		834	258			
26 scsd1	8.6666666743E+00	77	760	2388	760	1			77		760				
27 etamacro	-7.5571523337E+02	400	688	2409	80	24	48	80	272		426	135		82	45
28 agg	-3.5991767287E+07	488	163	2410	131	432	405	47	36		163				
29 bandm	-1.5862801845E+02	305	472	2494	165	118			305		472				
30 e226	-1.8751929066E+01	223	282	2578	189	99	185	5	33		282				
31 scfxm1	1.8416759028E+04	330	457	2589	23	116	143		187		457				
32 grow7	-4.7787811815E+07	140	301	2612	21	0			140		21	280			
33 standata	1.2576995000E+03	359	1183	3031	7	7	199		160		1063	104		16	
34 scrs8	9.0429695380E+02	490	1169	3182	847	77	59	47	384		1169				
35 beaconfd	3.3592485807E+04	173	262	3375	101	67	33		140		262				
36 boeing1	-3.3521356751E+02	351	384	3485	380	146	4	249	9	89	228	156			
37 shell	1.2088253460E+09	536	1775	3556	1344	2	2		534		1399	117		250	9
38 standmps	1.4060175000E+03	467	1075	3679	7	115	199		268		955	104		16	
39 stair	-2.5126695119E+02	356	467	3856	1	70	147		209		373	6	6	82	
40 degen2	-1.4351780000E+03	444	534	3978	471	243	223		221		534				
41 agg2	-2.0239252356E+07	516	302	4284	231	472	456		60		302				
42 agg3	1.0312115935E+07	516	302	4300	231	467	456		60		302				
43 scsd6	5.0500000077E+01	147	1350	4316	1350	9			147		1350				
44 ship04s	1.7987147004E+06	402	1458	4352	1458	263	40	8	354		1458				
45 seba	1.5711600000E+04	515	1028	4352	522	9		1	507	7	521	507			
46 tuff	2.9214776509E-01	333	587	4520	3	0	15	26	292		559	26	2		
47 forplan	-6.6421896127E+02	161	420	4563	353	15	50	20	90	1	396	21		3	
48 bnl1	1.9776295615E+03	643	1175	5121	1008	276	205	206	232		1175				
49 pilot4	-2.5811392589E+03	410	1000	5141	4	165	26	97	287		635	247	88	30	
50 scfxm2	3.6660261565E+04	660	914	5183	46	237	286		374		914				

TABLE I
Netlib Problem Data

PROBLEM		Optimal Value	Cons.	Vars.	Nonzeros Cons.	Nonzeros Obj.	Nonzeros RHS	Constraint Types Less	Constraint Types Greater	Constraint Types Equal	Constraint Types Range	Variable Types Nonneg.	Variable Types Box	Variable Types Free	Variable Types Fix	Variable Types Other
51	grow15	-1.0687094129E+08	300	645	5620	45	0			300		45	600			
52	perold	-9.3807552782E+03	625	1376	6018	8	215	40	90	495		951	266	88	64	7
53	fffff800	5.5567956482E+05	524	854	6227	8	204	93	81	350		854				
54	ship04l	1.7933245380E+06	402	2118	6332	2118	263	40	8	354		2118				
55	sctap2	1.7248071429E+03	1090	1880	6714	1410	521		620	470		1880				
56	ganges	-1.0958573613E+05	1309	1681	6912	109	491	25		1284		1277	397			7
57	ship08s	1.9200982105E+06	778	2387	7114	2387	413	72	8	698		2387				
58	sierra	1.5394362184E+07	1227	2036	7302	1950	781	633	66	528			2016		20	
59	scfxm3	5.4901254550E+04	990	1371	7777	69	358	429		561		1371				
60	ship12s	1.4892361344E+06	1151	2763	8178	2763	653	101	5	1045		2763				
61	grow22	-1.6083433648E+08	440	946	8252	66	0			440		66	880			
62	stocfor2	-3.9024408538E+04	2157	2031	8343	1149	8	888	126	1143		2031				
63	scsd8	9.0499999993E+02	397	2750	8584	2750	15			397		2750				
64	sctap3	1.4240000000E+03	1480	2480	8874	1860	682		860	620		2480				
65	pilotwe	-2.7201075328E+06	722	2789	9126	92	73	30	109	583		2335	294	80	78	2
66	maros	-5.8063743701E+04	846	1443	9614	392	42	399	124	323		1405			35	3
67	fit1p	9.1463780924E+03	627	1677	9868	1026	627			627		1278	399			
68	25fv47	5.5018458883E+03	821	1571	10400	727	287	305		516		1571				
69	czprob	2.1851966989E+06	929	3523	10669	3504	860	38	1	890		3294			229	
70	ship08l	1.9090552114E+06	778	4283	12802	4283	413	72	8	698		4283				
71	pilotnov	-4.4972761882E+03	975	2172	13057	72	332	151	123	701		1628	340		204	
72	nesm	1.4076036488E+07	662	2923	13288	700	542	94		480	88	1009	1508		175	231
73	fit1d	-9.1463780924E+03	24	1026	13404	1026	0	12	11	1			1026			
74	bnl2	1.8112365404E+03	2324	3489	13999	2125	441	482	515	1327		3489				
75	pilotja	-6.1131364654E+03	940	1988	14698	8	309	151	128	661		1250	339	88	311	
76	ship12l	1.4701879193E+06	1151	5427	16170	5427	653	101	5	1045		5427				
77	cycle	-5.2263930249E+00	1903	2857	20720	602	0	146	368	1389		2773	77	7		
78	80bau3b	9.8722419241E+05	2262	9799	21002	8061	346	35	2227			6244	2986		498	71
79	degen3	-9.8729400000E+02	1503	1818	24646	1584	594	786		717		1818				
80	truss	4.5881584719E+05	1000	8806	27836					1000		8806				
81	greenbea	-7.2555248130E+07	2392	5405	30877	622	0	107	86	2199		4996	290		103	16
82	greenbeb	-4.3022602612E+06	2392	5405	30877	622	0	107	86	2199		4978	291	4	115	17
83	d2q06c	1.2278421081E+05	2171	5167	32417	3257	874	664		1507		5167				
84	woodw	1.3044763331E+00	1098	8405	37474	4	34	9	4	1085		8405				
85	pilots	-5.5748972928E+02	1441	3652	43167	53	282	1191	17	233		2320	1040		203	89
86	fit2p	6.8464293294E+04	3000	13525	50284	10500	1500			3000		6025	7500			
87	stocfor3	-3.9976783944E+04	16675	15695	64875	9129	8	6866	980	8829		15695				
88	wood1p	1.4429024116E+00	244	2594	70215	1	2		1	243		2594				
89	pilot87	3.0171034733E+02	2030	4883	73152	652	274	1708	89	233		2971	1578		220	114
90	fit2d	-6.8464293294E+04	25	10500	129018	9000	0	10	14	1			10500			

TABLE II

Stability of Optimal Bases

PROBLEM		Reoptimized Basis Condition Number		Default Settings Objective Value	Reoptimized Objective Value	Relative Objective Error
		Unscaled	Scaled			
1	afiro	3.21E+01	3.13E+01	-4.647531428571E+02	-4.647531428571E+02	
2	sc50b	1.23E+02	1.23E+02	-7.000000000000E+01	-7.000000000000E+01	
3	sc50a	1.28E+02	9.52E+01	-6.457507705856E+01	-6.457507705856E+01	
4	sc105	4.83E+02	3.28E+02	-5.220206121171E+01	-5.220206121171E+01	
5	kb2	1.56E+06	2.87E+03	-1.749900129906E+03	-1.749900129906E+03	
6	adlittle	5.47E+03	4.35E+02	2.254949631624E+05	2.254949631624E+05	
7	scagr7	1.04E+04	4.24E+03	-2.331389824331E+06	-2.331389824331E+06	
8	stocfor1	5.07E+05	1.93E+02	-4.113197621944E+04	-4.113197621944E+04	
9	blend	1.33E+04	4.07E+02	-3.081214984583E+01	-3.081214984583E+01	
10	sc205	2.30E+03	1.75E+03	-5.220206121171E+01	-5.220206121171E+01	
11	recipe	4.82E+05	1.04E+02	-2.666160000000E+02	-2.666160000000E+02	
12	share2b	1.29E+06	7.49E+03	-4.157322407414E+02	-4.157322407414E+02	
13	vtpbase	1.85E+09	9.23E+04	1.298314624614E+05	1.298314624614E+05	
14	lotfi	1.05E+07	3.76E+04	-2.526470606188E+01	-2.526470606188E+01	
15	share1b	2.39E+07	3.97E+03	-7.658931857919E+04	-7.658931857919E+04	
16	boeing2	4.89E+06	5.34E+04	-3.150187280152E+02	-3.150187280152E+02	
17	scorpion	3.24E+03	2.38E+03	1.878124822738E+03	1.878124822738E+03	
18	bore3d	1.54E+07	6.43E+04	1.373080394208E+03	1.373080394208E+03	
19	scagr25	1.38E+04	4.18E+03	-1.475343306077E+07	-1.475343306077E+07	
20	sctap1	6.82E+04	1.68E+02	1.412250000000E+03	1.412250000000E+03	
21	capri	1.13E+06	3.84E+04	2.690012913768E+03	2.690012913768E+03	
22	brandy	1.24E+05	1.32E+03	1.518509896488E+03	1.518509896488E+03	
23	israel	1.53E+07	2.66E+04	-8.966448218630E+05	-8.966448218630E+05	
24	finnis	9.91E+04	3.66E+03	1.727910655956E+05	1.727910655956E+05	
25	gfrdpnc	1.94E+05	1.38E+05	6.902235999549E+06	6.902235999549E+06	
26	scsd1	7.65E+01	7.65E+01	8.666666674333E+00	8.666666674333E+00	
27	etamacro	5.53E+06	5.66E+03	-7.557152298914E+02	-7.557152333749E+02	4.610E-09
28	agg	5.08E+07	2.04E+05	-3.599176728658E+07	-3.599176728658E+07	
29	bandm	2.39E+05	5.16E+03	-1.586280184501E+02	-1.586280184501E+02	
30	e226	1.57E+07	1.11E+04	-1.875192906637E+01	-1.875192906637E+01	
31	scfxm1	2.88E+06	3.93E+03	1.841675902835E+04	1.841675902835E+04	
32	grow7	4.16E+02	3.64E+02	-4.778781181471E+07	-4.778781181471E+07	
33	standata	3.23E+04	8.53E+02	1.257699500000E+03	1.257699500000E+03	
34	scrs8	5.65E+05	2.78E+03	9.042969538008E+02	9.042969538008E+02	
35	beaconfd	1.04E+04	4.02E+01	3.359248580720E+04	3.359248580720E+04	
36	boeing1	5.86E+07	2.29E+05	-3.352135675071E+02	-3.352135675071E+02	
37	shell	1.20E+02	1.20E+02	1.208825346000E+09	1.208825346000E+09	
38	standmps	3.28E+04	2.73E+03	1.406017500000E+03	1.406017500000E+03	
39	stair	8.94E+04	2.16E+04	-2.512669511930E+02	-2.512669511930E+02	
40	degen2	3.80E+03	3.80E+03	-1.435178000000E+03	-1.435178000000E+03	
41	agg2	1.58E+05	6.71E+03	-2.023925235598E+07	-2.023925235598E+07	
42	agg3	3.43E+05	1.40E+04	1.031211593509E+07	1.031211593509E+07	
43	scsd6	9.31E+02	9.31E+02	5.050000007826E+01	5.050000007714E+01	2.218E-11
44	ship04s	2.29E+03	2.65E+02	1.798714700445E+06	1.798714700445E+06	
45	seba	3.78E+06	1.67E+05	1.571160000000E+04	1.571160000000E+04	
46	tuff	1.00E+08	7.63E+03	2.921477650936E-01	2.921477650936E-01	
47	forplan	8.68E+07	7.82E+04	-6.642189612722E+02	-6.642189612722E+02	
48	bnl1	2.12E+07	1.37E+06	1.977629561523E+03	1.977629561523E+03	
49	pilot4	5.48E+10	4.95E+06	-2.581139259543E+03	-2.581139258885E+03	2.549E-10
50	scfxm2	4.09E+06	3.95E+03	3.666026156500E+04	3.666026156500E+04	

TABLE II

Stability of Optimal Bases

PROBLEM		Reoptimized Basis Condition Number Unscaled	Scaled	Default Settings Objective Value	Reoptimized Objective Value	Relative Objective Error
51	grow15	2.76E+03	2.68E+03	-1.068709412936E+08	-1.068709412936E+08	
52	perold	1.37E+10	4.54E+06	-9.380755278232E+03	-9.380755278235E+03	3.198E-13
53	fffff800	4.72E+10	1.39E+07	5.556795648175E+05	5.556795648175E+05	
54	ship04l	8.73E+02	7.13E+01	1.793324537970E+06	1.793324537970E+06	
55	sctap2	5.20E+04	2.11E+02	1.724807142857E+03	1.724807142857E+03	
56	ganges	1.98E+05	1.32E+05	-1.095857361293E+05	-1.095857361293E+05	
57	ship08s	1.03E+04	1.48E+02	1.920098210535E+06	1.920098210535E+06	
58	sierra	1.00E+15	1.23E+02	1.539436218363E+07	1.539436218363E+07	
59	scfxm3	4.09E+06	3.95E+03	5.490125454975E+04	5.490125454975E+04	
60	ship12s	1.01E+04	1.44E+02	1.489236134406E+06	1.489236134406E+06	
61	grow22	2.32E+03	2.15E+03	-1.608343364826E+08	-1.608343364826E+08	
62	stocfor2	9.55E+06	7.16E+03	-3.902440853788E+04	-3.902440853788E+04	
63	scsd8	1.67E+03	1.67E+03	9.049999999255E+02	9.049999999255E+02	
64	sctap3	2.20E+04	1.70E+02	1.424000000000E+03	1.424000000000E+03	
65	pilotwe	1.39E+10	1.49E+08	-2.720107532765E+06	-2.720107532803E+06	1.397E-11
66	maros	6.93E+09	1.19E+05	-5.806374370113E+04	-5.806374370113E+04	
67	fit1p	1.22E+08	6.42E+05	9.146378092421E+03	9.146378092421E+03	
68	25fv47	1.94E+06	2.03E+04	5.501845888287E+03	5.501845888287E+03	
69	czprob	1.48E+04	5.82E+02	2.185196698857E+06	2.185196698857E+06	
70	ship08l	1.73E+04	3.21E+02	1.909055211389E+06	1.909055211389E+06	
71	pilotnov	4.31E+13	1.28E+08	-4.497276188219E+03	-4.497276188219E+03	
72	nesm	1.34E+06	2.51E+06	1.407603648756E+07	1.407603648756E+07	
73	fit1d	1.08E+06	1.10E+03	-9.146378092421E+03	-9.146378092421E+03	
74	bnl2	5.66E+07	3.02E+06	1.811236540359E+03	1.811236540359E+03	
75	pilotja	5.37E+12	2.28E+09	-6.113136465533E+03	-6.113136465428E+03	1.718E-11
76	ship12l	1.01E+04	1.44E+02	1.470187919329E+06	1.470187919329E+06	
77	cycle	1.25E+08	5.19E+05	-5.226393024894E+00	-5.226393024894E+00	
78	80bau3b	2.28E+04	6.01E+02	9.872241924091E+05	9.872241924091E+05	
79	degen3	5.35E+04	5.35E+04	-9.872940000000E+02	-9.872940000000E+02	
80	truss	8.57E+02	8.57E+02	4.588158471856E+05	4.588158471856E+05	
81	greenbea	4.53E+08	3.21E+07	-7.255524812985E+07	-7.255524812985E+07	
82	greenbeb	2.96E+06	1.58E+05	-4.302260261207E+06	-4.302260261207E+06	
83	d2q06c	7.38E+08	1.46E+06	1.227842108142E+05	1.227842108142E+05	
84	woodw	8.30E+04	1.55E+02	1.304476333084E+00	1.304476333084E+00	
85	pilots	3.30E+08	2.96E+07	-5.574897061534E+02	-5.574897292840E+02	4.149E-08
86	fit2p	1.24E+09	2.77E+07	6.846429329383E+04	6.846429329383E+04	
87	stocfor3	2.28E+08	2.49E+05	-3.997678394365E+04	-3.997678394365E+04	
88	wood1p	2.32E+04	4.65E+01	1.442902411573E+00	1.442902411573E+00	
89	pilot87	5.32E+08	1.43E+07	3.017103666613E+02	3.017103473331E+02	6.406E-08
90	fit2d	1.98E+04	2.77E+03	-6.846429329383E+04	-6.846429329383E+04	

TABLE III

Residuals for Optimal Bases

	Default Settings Optimization								Reoptimized							
	Bound		Reduced-Cost		AX-b		c_B'-pi'B		Bound		Reduced-Cost		AX-b		c_B'-pi'B	
	Infeasibilities		Infeasibilities		Max. Residual		Max. Residual		Infeasibilities		Infeasibilities		Max. Residual		Max. Residual	
PROBLEM	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.
1 afiro					1E-14	1E-14	6E-17	6E-17					2E-14	2E-14	6E-17	6E-17
2 sc50b					1E-13		1E-16						1E-13		1E-16	
3 sc50a					2E-13	1E-13	1E-16	1E-16					3E-14	1E-14	1E-16	1E-16
4 sc105					9E-13	5E-13	1E-16	1E-16					3E-14	1E-14	1E-16	1E-16
5 kb2					8E-12	7E-14	1E-14	1E-14					2E-12	2E-14	2E-15	2E-15
6 adlittle			1E-13	1E-13	2E-13	1E-13	2E-13	2E-13			3E-13	3E-13	1E-13	1E-13	2E-13	2E-13
7 scagr7					1E-12	6E-13	5E-13	5E-13					1E-12	6E-13	5E-13	5E-13
8 stocfor1					5E-13	7E-15	9E-14	9E-14					5E-13	7E-15	1E-13	1E-13
9 blend					2E-13	4E-15	1E-15	1E-15					1E-14	4E-15	2E-15	2E-15
10 sc205					1E-12	6E-13	1E-16	1E-16					2E-13	1E-13	1E-16	1E-16
11 recipe																
12 share2b			4E-17	4E-15	4E-13	4E-15	7E-14	7E-14			9E-16	9E-16	4E-13	4E-15	6E-14	6E-14
13 vtpbase	1E-14	5E-17			3E-11	6E-14	4E-12	4E-12	1E-14	5E-17			3E-11	6E-14	7E-12	7E-12
14 lotfi	2E-13	4E-15			9E-10	9E-13	8E-17	8E-17	1E-12	1E-14			1E-12	1E-14	8E-17	8E-17
15 share1b					8E-10	9E-13	7E-15	7E-15					4E-10	2E-12	1E-14	1E-14
16 boeing2			9E-18	9E-16	7E-13	5E-13	1E-15	1E-15	9E-16	9E-16			6E-13	2E-13	2E-15	2E-15
17 scorpion	2E-16	2E-16			4E-16	4E-16	1E-13	1E-13	5E-17	8E-17			4E-16	4E-16	1E-13	1E-13
18 bore3d					6E-13	5E-13	2E-13	2E-13			6E-17	1E-16	6E-13	2E-13	9E-15	9E-15
19 scagr25					2E-12	1E-12	1E-12	1E-12					2E-12	9E-13	1E-12	1E-12
20 sctap1	2E-15	2E-16	7E-16	1E-14	9E-15	2E-16	7E-15	1E-14	2E-16	2E-16	6E-16	1E-14	2E-14	2E-16	7E-15	7E-15
21 capri					2E-12	2E-12	9E-14	9E-14	1E-14	1E-14			5E-13	5E-13	3E-14	4E-14
22 brandy					4E-13	1E-14	2E-14	2E-14					4E-13	1E-14	3E-15	3E-15
23 israel			2E-15	9E-15	1E-10	2E-12	9E-14	9E-14			4E-15	7E-15	1E-10	9E-13	3E-13	3E-13
24 finnis	7E-15	2E-15			5E-13	1E-13	3E-13	3E-13	7E-15	2E-15	3E-14	9E-14	4E-13	2E-13	3E-13	3E-13
25 gfrdpnc	2E-12	1E-15			5E-12	2E-12	7E-12	7E-12	1E-12	1E-15			5E-12	2E-12	7E-12	1E-11
26 scsd1					1E-16		1E-15		1E-17				1E-16		5E-15	
27 etamacro	3E-16	3E-16	7E-07	7E-07	1E-12	2E-13	3E-14	3E-14	3E-17	3E-17			1E-12	2E-13	3E-14	3E-14
28 agg					5E-10	9E-10	6E-14	6E-14					5E-10	9E-10	2E-13	2E-13
29 bandm					3E-12	3E-14	2E-13	2E-13					4E-13	6E-15	2E-15	2E-15
30 e226					1E-13	2E-15	1E-15	1E-15					7E-14	2E-15	9E-16	9E-16
31 scfxm1	2E-15	2E-15			2E-12	7E-13	9E-15	9E-15	2E-15	2E-15			3E-12	2E-13	3E-15	3E-15
32 grow7			2E-11	2E-11	2E-08	2E-08	3E-13	3E-13					2E-10	2E-10	2E-15	2E-15
33 standata					5E-13	2E-14	2E-15	2E-15					5E-13	2E-14	2E-15	2E-15
34 scrs8	2E-16	2E-16			3E-14	9E-16	9E-13	9E-13	2E-16	2E-16			6E-14	2E-15	4E-13	4E-13
35 beaconfd					3E-11	3E-13	2E-15	2E-15					3E-11	3E-13	2E-15	2E-15
36 boeing1			5E-18	8E-17	6E-12	3E-12	9E-15	9E-15			9E-18	2E-16	6E-12	9E-13	3E-15	3E-15
37 shell																
38 standmps					2E-13	5E-15	2E-15	2E-15					2E-13	5E-15	4E-15	4E-15
39 stair					2E-11	6E-12	6E-14	7E-14					2E-13	9E-14	2E-15	2E-15
40 degen2	2E-16				9E-16		7E-15		2E-16				9E-16		1E-14	
41 agg2					7E-11	3E-10	1E-14	1E-14					1E-10	5E-10	1E-14	1E-14
42 agg3					6E-11	5E-10	1E-14	1E-14					1E-10	5E-10	1E-14	1E-14
43 scsd6			2E-08		2E-16		4E-15						3E-16		2E-15	
44 ship04s	2E-14	2E-14	1E-13	1E-13	5E-14	5E-14			2E-14	2E-14			5E-14	5E-14		
45 seba					2E-12	6E-14							2E-12	6E-14		
46 tuff					2E-12	2E-14	9E-18	9E-18					5E-12	4E-14	2E-18	2E-18
47 forplan					2E-12	9E-13	1E-14	1E-14					2E-12	9E-13	1E-14	1E-14
48 bnl1	3E-17	3E-17	4E-14	3E-13	4E-13	6E-14	2E-13	2E-13	1E-12	1E-12	1E-12	9E-12	3E-13	6E-14	1E-13	1E-13
49 pilot4	2E-07	5E-08	1E-13	1E-13	2E-07	2E-09	4E-13	4E-13	9E-13	1E-15			3E-07	8E-10	1E-13	1E-13
50 scfxm2	2E-15	2E-15	2E-15	4E-16	6E-12	1E-12	6E-14	6E-14	2E-15	2E-15			4E-12	7E-13	1E-14	1E-14

TABLE III
Residuals for Optimal Bases

	Default Settings Optimization								Reoptimized							
	Bound Infeasibilities		Reduced-Cost Infeasibilities		AX-b Max. Residual		c_B'-pi'B Max. Residual		Bound Infeasibilities		Reduced-Cost Infeasibilities		AX-b Max. Residual		c_B'-pi'B Max. Residual	
PROBLEM	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.	Unscl.	Scl.
51 grow15			7E-12	7E-12	3E-09	3E-09	6E-12	7E-12			2E-13	2E-13	2E-09	2E-09	2E-13	2E-13
52 perold	4E-12	9E-15	1E-13	1E-13	2E-06	2E-09	1E-11	1E-11	4E-12	9E-15	8E-14	1E-13	3E-09	3E-12	1E-13	1E-13
53 fffff800					1E-10	2E-11	3E-11	3E-11					2E-10	3E-11	3E-11	3E-11
54 ship04l			4E-13	4E-13	2E-14	3E-14					4E-13	4E-13	2E-14	3E-14		
55 sctap2			5E-16	1E-14	9E-15	2E-16	7E-15	1E-14	6E-16	6E-16	4E-16	7E-15	9E-15	2E-16	1E-14	1E-14
56 ganges					1E-11	1E-11	6E-14	6E-14			1E-13	1E-13	1E-11	1E-11	8E-14	8E-14
57 ship08s	5E-16	5E-16			2E-14	2E-14			5E-16	5E-16			2E-14	2E-14		
58 sierra	2E-13	2E-13	6E-07	6E-07	9E-13	9E-13	4E-12	4E-12	2E-13	2E-13	4E-12	4E-12	9E-13	9E-13	4E-12	4E-12
59 scfxm3	2E-15	2E-15	2E-15	7E-16	4E-12	7E-13	1E-13	1E-13	2E-15	2E-15	2E-15	7E-16	7E-12	7E-13	1E-14	1E-14
60 ship12s	2E-15	3E-14			3E-14	6E-14	9E-13	9E-13	2E-15	4E-14			3E-14	6E-14	9E-13	9E-13
61 grow22			5E-11	5E-11	3E-07	3E-07	8E-12	9E-12			7E-12	7E-12	2E-08	2E-08	5E-12	7E-12
62 stocfor2	7E-16	7E-16			5E-12	2E-14	6E-14	6E-14	7E-16	7E-16			2E-12	1E-14	9E-14	9E-14
63 scsd8	4E-16				4E-15		1E-14		2E-16				2E-15		2E-14	
64 sctap3			4E-16	7E-15	9E-15	2E-16	2E-14	2E-14			4E-16	7E-15	9E-15	2E-16	1E-14	1E-14
65 pilotwe	7E-12	2E-15			4E-07	5E-09	6E-10	6E-10	7E-12	2E-15			4E-05	8E-09	1E-09	1E-09
66 maros	2E-10	1E-14	5E-13	5E-13	5E-10	5E-12	3E-12	3E-12	8E-13	1E-13			9E-10	9E-13	2E-14	3E-14
67 fit1p					5E-14	2E-16	2E-12	2E-12					5E-14	2E-16	2E-12	2E-12
68 25fv47					1E-11	1E-11	2E-13	2E-13					2E-12	3E-13	2E-14	2E-14
69 czprob					6E-13	6E-13	3E-14	3E-14					2E-12	2E-12	3E-14	3E-14
70 ship08l	4E-15	4E-15			4E-14	4E-14	9E-13	9E-13	4E-15	4E-15			2E-14	2E-14	9E-13	9E-13
71 pilotnov	1E-16	1E-16			6E-09	1E-10	3E-16	3E-16	2E-12	2E-15			5E-07	4E-09	1E-16	1E-16
72 nesm	1E-08	1E-08	2E-08	2E-07	1E-11	7E-13	2E-12	2E-12	9E-13	8E-14	7E-11	7E-11	3E-12	9E-13	8E-13	8E-13
73 fit1d					7E-13	9E-16	7E-15	7E-15					7E-13	9E-16	7E-15	7E-15
74 bnl2	4E-14	4E-14	7E-07	7E-07	2E-13	8E-14	1E-14	1E-14	2E-14	2E-14	1E-15	4E-15	2E-13	1E-13	1E-14	1E-14
75 pilotja	7E-12	3E-16	3E-15	3E-15	7E-05	3E-09	2E-12	2E-12	7E-12	3E-16	2E-13	2E-13	2E-04	6E-09	2E-11	2E-11
76 ship12l	3E-15	3E-15			5E-14	6E-14	5E-13	5E-13	3E-15	3E-15			5E-14	6E-14		
77 cycle					3E-12	3E-14	3E-17	3E-17	3E-13	7E-15	4E-19	4E-19	2E-12	6E-14	3E-17	3E-17
78 80bau3b	2E-13	1E-13			2E-12	2E-12	3E-14	3E-14	2E-13	1E-13			2E-12	2E-12	1E-14	1E-14
79 degen3	4E-15				1E-14		4E-15		2E-15		4E-16		7E-15		4E-15	
80 truss	4E-15				2E-13		2E-12		4E-15				2E-13		2E-12	
81 greenbea	2E-13	2E-13	1E-12	1E-12	1E-08	1E-08	2E-11	2E-11	2E-13	2E-13	1E-12	1E-12	1E-08	1E-08	3E-11	3E-11
82 greenbeb	1E-13	1E-13	7E-17	1E-16	7E-12	7E-12	4E-12	4E-12	1E-13	1E-13	3E-13	3E-13	3E-12	3E-12	4E-12	4E-12
83 d2q06c			2E-15	2E-15	8E-10	7E-12	2E-13	2E-13			6E-16	2E-15	5E-10	9E-13	2E-14	2E-14
84 woodw					1E-13	1E-16	3E-16	3E-16					1E-13	1E-16	3E-16	3E-16
85 pilots	3E-08	2E-08	1E-08	5E-08	6E-10	8E-11	2E-12	2E-12	1E-13	2E-14	2E-16	2E-16	7E-10	7E-11	9E-14	9E-14
86 fit2p	1E-14	6E-16			7E-14	4E-15	3E-11	3E-11	1E-14	6E-16			7E-14	4E-15	3E-11	3E-11
87 stocfor3	1E-15	1E-15			2E-11	7E-14	4E-14	4E-14	3E-15	3E-15			9E-12	3E-14	1E-13	1E-13
88 wood1p	4E-14	4E-17			1E-13	3E-16	4E-15	4E-15	2E-14	2E-17			9E-14	7E-16	3E-15	3E-15
89 pilot87	2E-12	1E-13	2E-07	4E-07	6E-08	6E-09	2E-12	2E-12	2E-12	1E-13			1E-09	1E-10	9E-15	1E-14
90 fit2d					1E-13	4E-16	1E-14	1E-14					1E-13	4E-16	1E-14	1E-14

Part II: The Initial Basis

5 Constructing an Initial Basis

The simplex method requires as input a feasible basis. If no such basis is available, it is standard to begin by constructing some sort of auxiliary or "phase I" problem. This auxiliary problem is then solved, and the resulting basis (in the case of feasibility) is used as the starting basis to solve the original problem.

In this section four alternate initial bases are described. The first is the classical all "artificial" basis, essentially that presented in most texts on linear programming. The next two bases, the feasible slack and slack, can be viewed as the natural next steps in an attempt to eliminate the use of artificial variables. Indeed, they were the intermediate steps that led to the development of the initial basis that is currently used in CPLEX. That basis is described last.

Suppose that (B, N_l, N_u) satisfies all the conditions for a basis, except feasibility. For $j = 1, \ldots, n$, define

$$p_j(x) = \begin{cases} x - u_j & \text{if} \quad x > u_j, \\ 0 & \text{if} \quad l_j \leq x \leq u_j, \text{ and} \\ l_j - x & \text{if} \quad l_j > x. \end{cases} \tag{3}$$

The associated *phase I* problem is piecewise linear: [5]

$$\begin{array}{ll} \text{minimize} & \sum_{j=1}^{n} p_j(x_j) \\ \text{s.t.} & Ax = b \end{array} \tag{4}$$

The basis B is obviously feasible for (4), and (1) has a feasible solution if and only if the optimal objective value of (4) is 0. It should be noted that the initial values of the basic variables in the slack and CPLEX bases, described below, may violate their bounds, and so dictate the need for viewing the phase I problem as piecewise linear. On the other hand, the artificial and feasible slack bases result in initial optimization problems that are ordinary LP's.

All Artificial Basis

The simplest approach to finding an initial basis is to begin with an all "artificial" basis. Let

$$\begin{array}{lcl} N & = & \{1, \ldots, n\}, \\ N_l & = & \{j \in N \backslash (N_{fr} \cup N_{fx}) : |l_j| \leq |u_j|\}, \\ N_u & = & \{j \in N \backslash (N_{fr} \cup N_{fx}) : |l_j| > |u_j|\}, \text{ and} \\ \tilde{b} & = & b - A_N X_N, \end{array} \tag{5}$$

[5]While (4) is not an LP, it is easy to see that all of the definitions in section 2 can be applied to it.

and consider the problem

$$\begin{array}{ll} \text{minimize} & e^T z \\ \text{s.t.} & Ax + Dz = b \\ & l \leq x \leq u \\ & z \geq 0 \end{array} \tag{6}$$

where $z^T = (x_{n+1}, \ldots, x_{n+m})$, $e^T = (1, \ldots, 1)$ and $D = \operatorname{diag}(\delta_1, \ldots, \delta_m)$ with

$$\delta_i = \begin{cases} +1 & \text{if } \bar{b}_i \geq 0, \text{ and} \\ -1 & \text{otherwise.} \end{cases}$$

The variables z are called *artificial* variables. Clearly (1) is feasible if and only if the optimum value of (6) is 0.

The *artificial basis* for (6) is (B, N_l, N_u) where $B = (n+1, \ldots, n+m)$, and N_l and N_u are as given in (5). B is clearly feasible for (6).

Feasible Slack Basis

The typical LP, as specified by the user, contains some inequality constraints:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{s.t.} & A_1 x \leq b_1 \\ & A_2 x \geq b_2 \\ & A_3 x = b_3 \\ & l \leq x \leq u \end{array} \tag{7}$$

Assume that b_i is an m_i-vector for $i = 1, 2, 3$ $(m = m_1 + m_2 + m_3)$. Problem (7) is converted to form (1) by adding *slack variables* $s_1^T = (x_{n+1}, \ldots, x_{n+m_1})$ and $s_2^T = (x_{n+m_1+1}, \ldots, x_{n+m_1+m_2})$:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{s.t.} & A_1 x + s_1 = b_1 \\ & A_2 x - s_2 = b_2 \\ & A_3 x \qquad = b_3 \\ & \quad l \leq x \leq u \\ & s_1 \geq 0, s_2 \geq 0 \end{array}$$

For this problem, the first step in constructing a *feasible slack basis* is to construct the artificial basis, noting that all slack variables will initially be in N_l. Then, for $i \in \{1, \ldots, m_1\}$ if $\delta_i = 1$, and for $i \in \{m_1 + 1, \ldots, m_1 + m_2\}$ if $\delta_i = -1$, the artificial $x_{n+m_1+m_2+i}$ variable is replaced in B by the slack x_{m+i}. In other words, available slacks are used so long as they are initially nonnegative.

Slack Basis

In this approach, all available slacks are used, independent of feasibility. Consider the problem

$$\begin{array}{ll} \text{minimize} & \sum_{j=1}^{n+m} p_j(x_j) \\ \text{s.t.} & A_1 x + s_1 = b_1 \\ & A_2 x - s_2 = b_2 \\ & A_3 x + z \;= b_3 \end{array}$$

where the p_j are given by (3), using the bounds $s_1 \geq 0$, $s_2 \geq 0$ and $z = 0$ for the slacks and artificials, respectively. The initial basis is (B, N_l, N_u) where $B = (n+1, \ldots, n+m)$, and N_l and N_u are as given in (5).

CPLEX Basis

The procedure described below is more complicated than the three described above. However, its implementation is quite simple. The essential idea is to construct a sparse, well-behaved basis, with as much freedom as possible, and as few artificials as possible. First, a preference order for the variables is constructed; this order is then used to construct the basis. As the computational results will show, using the initial basis constructed in this way can greatly reduce the number of iterations, especially for "easy problems." For harder problems, it is generally less effective.

It is perhaps natural to think that just the opposite would be true, that the starting basis should be *more* effective for harder problems. Indeed, precisely that consideration is the main motivation for developing so-called "crash" procedures, some of which can be remarkably effective in dealing with particular problem structures, such as those arising in multiperiod models. However, in the opinion of this author, the CPLEX initial basis is best viewed not as a crash, but simply as a default starting basis. In general, trying to *guess* a correct basis for a problem with unknown origins can have disastrous consequences. The goal of the CPLEX basis is not so much to try to find variables that are likely to be in any optimal basis, but to avoid the work of removing artificial variables, and avoid the effect that their restrictive bounds have on the optimization.

Assume that the given problem has the form (7) and has been converted to form (1) by adding slack variables. The preference order for variables is determined as follows. Define the following sets, where C_i will be "preferred" to C_{i+1} $(i = 1, 2, 3)$:

$$\begin{aligned} C_1 &= \{n+1, \ldots, n+m_1+m_2\}, \\ C_2 &= \{j : x_j \text{ free}\}, \\ C_3 &= \{j \leq n : \text{ exactly one of } l_j \text{ and } u_j \text{ is finite}\}, \text{ and} \\ C_4 &= \{j : -\infty < l_j, u_j < +\infty\}. \end{aligned}$$

Every variable falls into one of the above sets. Note that C_1 is just the set of indices of the slack variables. Slack variables are preferred to free variables because of the sparsity and numerical properties of the corresponding matrix columns.

For $j \in \{1, \ldots, n + m_1 + m_2\}$, define a penalty $\bar{q}_j$ by

$$\bar{q}_j = \begin{cases} 0 & \text{if } j \in C_2, \\ l_j & \text{if } j \in C_3 \text{ and } u_j = +\infty, \\ -u_j & \text{if } j \in C_3 \text{ and } l_j = -\infty, \text{ and} \\ l_j - u_j & \text{if } j \in C_4. \end{cases}$$

Let $\gamma = \max\{|c_j| : 1 \leq j \leq n\}$ and define

$$c_{max} = \begin{cases} 1000\gamma & \text{if } \gamma \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

Finally, for $j \in \{1, \ldots, n\}$, define

$$q_j = \bar{q}_j + c_j / c_{max}.$$

The variables within C_2, C_3 and C_4 are sorted in ascending order of q_j value and the lists are concatenated into a single ordered set $C = (j_1, \ldots, j_n)$. The sorting has the effect of placing the variables with the most "freedom" at the start, or preferred-end, of the list, using the objective function to break ties.

The basis B can now be constructed. After the construction is complete, the indices in $N \backslash (B \cup N_{fr} \cup N_{fx})$ are assigned to N_l and N_u according to (5). As previously noted, the constraint matrix is always scaled by CPLEX. The important feature of the scaling is that, after scaling, the maximum absolute value in every non-zero row and column is 1.

Step 1: For $i = 1, \ldots, m_1 + m_2$, set

$$\begin{aligned} I_i &\leftarrow 1, \text{ and} \\ r_i &\leftarrow 1. \end{aligned}$$

Set $B \leftarrow \{n+1, \ldots, n+m_1+m_2\}$. For $i = m_1 + m_2 + 1, \ldots, m$, set $I_i \leftarrow 0$ and $r_i \leftarrow 0$. For $i = 1, \ldots, m$ set $v_i \leftarrow +\infty$.

Step 2: Let $C = (j_1, \ldots, j_n)$. For $k = 1, \ldots, n$, apply the following procedure:

(a) Let $\alpha = \max\{|A_{lj_k}| : r_l = 0\}$. If $\alpha \geq 0.99$, let l' be such that $\alpha = |A_{l'j_k}|$ and $r_{l'} = 0$. Set

$$\begin{aligned} B &\leftarrow B \cup \{j_k\}, \\ I_{l'} &\leftarrow 1, \\ v_{l'} &\leftarrow \alpha, \text{ and} \\ r_l &\leftarrow r_l + 1 \text{ for all } l \text{ such that } |A_{lj_k}| \neq 0. \end{aligned}$$

Continue to the next k.

(b) If $|A_{lj_k}| > 0.01v_l$ for some l, continue to the next k; otherwise, let $\alpha = \max\{|A_{lj_k}| : I_l = 0\}$. If $\alpha = 0$, continue to the next k; otherwise, let l' be such that $I_{l'} = 0$ and $\alpha = |A_{l'j_k}|$. Set

$$\begin{array}{lcl} B & \leftarrow & B \cup \{j_k\}, \\ I_{l'} & \leftarrow & 1, \\ v_{l'} & \leftarrow & \alpha, \text{ and} \\ r_l & \leftarrow & r_l + 1 \text{ for all } l \text{ such that } |A_{lj_k}| \neq 0. \end{array}$$

Step 3: For $i = m_1 + m_2 + 1, \ldots, m$, if $I_i = 0$, set $B \leftarrow B \cup \{n + m_1 + m_2 + i\}$—that is, add an artificial variable to cover each remaining uncovered row. □

The v values in the above construction can be thought of as pseudo pivot values, and the condition that $|A_{lj_i}| \leq 0.01v_l$ in Step 2(b) as enforcing an approximate lower triangularity. Indeed, if 0.01 is replaced by 0, and v is initialized to any finite number for the slacks (Step 1), then the resulting matrix $\boldsymbol{B}$ is lower triangular. In a similar vein, if v is initialized to 1 for the slacks, and 0.99 is replaced by 1 in Step 2(a), then, it is easy to prove that *all* v values are 1.

6 Computational Results

All tests were run on a 40 Mhz SPARCstation 2. CPLEX is written entirely in C and was compiled using the flags '-O4 -cg89 -libmil -dalign' and the bundled Sun C compiler.

The CPLEX procedure for constructing an initial basis may be summarized as follows. It begins by placing all available slack variables in the basis. The columns of the constraint matrix are then ordered based on the "distance" between the bounds. Using this ordering, a heuristic factorization routine is applied (Step 2 at the end of the last section) to find "pivot" elements covering equality rows, those rows not covered by the slacks.

The inclusion of all slacks in the initial basis is motivated by the sparsity and "numerical stability" of these columns together with the expectation that a significant fraction will be in the optimal basis. The results in Table IV show that, for the netlib problems an average of 52.9% of the slacks were used in the optimal bases. For only six of the problems, including pilots and pilot87, was the percentage under 10.

The heuristic factorization was very successful in eliminating artificial variables. An average of only 13.1% of equality rows were covered by artificials in the initial bases. Almost none of the artificials remained in the optimal bases. One exception was 'cycle', where 130 artificial variables remained in the optimal basis, presumably due to redundancy among the equality constraints.

Each of the 90 netlib problems was solved using each of the four types of initial bases introduced in section 4: artificial (A), feasible slack (FS), slack (S) and CPLEX

(C). All runs were made in batch mode on a standalone system with no other user-jobs running. Default settings were used throughout in CPLEX. Only the initial basis was varied.

The timings reported are the user times returned by the 'times' library function. Totals for phase I iterations, total iterations and run times are reported at the bottom of the table. These numbers are dominated by several of the more difficult problems, and so are not as useful as one might hope. However, they do clearly indicate that the artificial basis is inferior to the CPLEX basis. As one might expect, the biggest part of this advantage comes in phase I. In addition, the improvement in total iterations is significantly greater than the improvement in run time—the CPLEX initial basis, though relatively sparse, is certainly more expensive to handle than an identity basis.

In an attempt to equalize the effect of the difficult problems, two kinds of ratios were added to Table V: ratios of the total iterations required by each of the three simpler bases divided by the iterations required by the CPLEX basis, and corresponding ratios for run times. Ratios for phase I iterations would not have been meaningful. Averages are presented at the bottom of the table. These numbers can be misleading if a sizeable number of the ratios are significantly less than 1.0. That, however, is not the case here. The CPLEX basis leads to slower solution times in 20 of 90 cases. Of those, 12 are by 10% or less and only 1 by more than 30% . The average improvement of CPLEX over the S basis is about 35%.

For problems that are difficult, the CPLEX basis is typically not a significant win. Ignoring bounds, a basis has size equal to the number of rows. Thus, a number of iterations approximating the number of rows can be interpreted as good performance: With no reasonable expectation that we can guess optimal columns in advance, it will take that many iterations just to pivot in the columns of an optimal basis.

Of the larger problems, stocfor3 is easy in the above sense. Even though the linear algebra is not cheap, the number of iterations is less than the number of rows, and the CPLEX basis is a significant improvement. On the other hand, greenbea, greenbeb, d2q06c, pilots, fit2p and pilot87 are not easy. The problem fit2d is not a good candidate for this discussion since a large number of variables must be moved to bounds, and the procedure used to select initial values for nonbasic variables was the same for each of the basis selection procedures.

TABLE IV

Artificials and Slacks in Optimal Bases

PROBLEM		Constraints Eq.	Constraints Ineq.	Initial Basis Artifs	Optimal Basis Slacks	Optimal Basis Artifs
1	afiro	8	19		7	
2	sc50b	20	30	1	2	
3	sc50a	20	30	1	4	
4	sc105	45	60	1	10	
5	kb2	16	27	3	16	
6	adlittle	15	41	4	10	
7	scagr7	84	45	7	32	
8	stocfor1	63	54	15	38	
9	blend	43	31	3	15	
10	sc205	91	114	1	9	
11	recipe*	63	24	15	22	7
12	share2b	13	83	2	43	
13	vtpbase	55	143	28	102	
14	lotfi	95	58	6	45	
15	share1b	89	28	27	23	
16	boeing2	23	143		87	
17	scorpion	280	108	73	16	
18	bore3d	214	19	36	13	6
19	scagr25	300	171	29	120	
20	sctap1	120	180		99	
21	capri	142	129	31	44	
22	brandy*	139	54	52	31	3
23	israel		174		105	
24	finnis	47	450		178	
25	gfrdpnc	548	68	113	17	1
26	scsd1	77		18		
27	etamacro	272	128	49	26	2
28	agg	36	452	6	408	
29	bandm	305		54		
30	e226	33	190	2	78	
31	scfxm1	187	143	39	75	
32	grow7	140		39		
33	standata	160	199	2	177	
34	scrs8	384	106	22	23	
35	beaconfd	140	33	1	33	
36	boeing1	9	253		129	
37	shell	534	2	99	2	1
38	standmps	268	199	22	176	
39	stair	209	147	16	6	
40	degen2	221	223	25	101	2
41	agg2	60	456	3	387	
42	agg3	60	456	7	386	
43	scsd6	147		38		
44	ship04s*	312	48	46	29	
45	seba	507	1	1	1	
46	tuff*	261	41	50	37	
47	forplan	90	70	11	47	
48	bnl1*	231	411	31	149	
49	pilot4	287	123	24	22	
50	scfxm2	374	286	73	148	

TABLE IV

Artificials and Slacks in Optimal Bases

PROBLEM	Constraints Eq.	Constraints Ineq.	Initial Basis Artif's	Optimal Basis Slacks	Optimal Basis Artif's
51 grow15	300		81		
52 perold	495	130	82	25	
53 fffff800*	350	174	17	99	2
54 ship04l*	312	48	46	32	
55 sctap2	470	620		379	
56 ganges	1284	25	254	24	
57 ship08s*	632	80	74	47	
58 sierra*	523	699	123	634	10
59 scfxm3	561	429	109	218	
60 ship12s*	936	106	133	50	
61 grow22	440		130		
62 stocfor2	1143	1014	147	644	
63 scsd8	397		72		
64 sctap3	620	860		516	
65 pilotwe	583	139	48	24	
66 maros*	322	522	20	252	
67 fit1p	627				
68 25fv47*	516	305	117	172	1
69 czprob	890	39	10	22	
70 ship08l*	632	80	74	53	1
71 pilotnov*	677	274	182	163	9
72 nesm	480	94	8	58	
73 fit1d	1	23		12	
74 bnl2	1327	997	212	543	
75 pilotja*	645	279	154	149	
76 ship12l*	936	106	121	46	
77 cycle*	1376	514	279	416	130
78 80bau3b		2262		133	
79 degen3	717	786	33	321	2
80 truss	1000		148		
81 greenbea*	2196	193	206	129	
82 greenbeb*	2196	193	202	122	3
83 d2q06c*	1507	664	153	382	2
84 woodw*	1085	13	2	9	1
85 pilots	233	1208	11	100	
86 fit2p	3000				
87 stocfor3	8829	7846		4532	
88 wood1p*	243	1	132		1
89 pilot87	233	1797	17	150	
90 fit2d	1	24		5	
Avg. % artif's/eq. rows in initial basis			13.1%		
Avg. % slacks in optimal basis				52.9%	
*Empty rows removed					

TABLE V

Comparison of Initial Bases:

Artificial (A), Feasible Slack (FS), Slack (S), CPLEX (C)

	Phase I Iterations				Total Iterations				Run Times (seconds)				Total Iterations Ratios			Run Time Ratios		
PROBLEM	A	FS	S	C	A	FS	S	C	A	FS	S	C	A/C	FS/C	S/C	A/C	FS/C	S/C
1 afiro	30	7	7	1	31	18	18	10	0.0	0.0	0.0	0.0	3.10	1.80	1.80	1.64	1.14	1.14
2 sc50b	52	0	0	0	70	48	48	32	0.1	0.1	0.1	0.1	2.19	1.50	1.50	1.33	1.18	1.18
3 sc50a	43	0	0	0	68	44	44	28	0.1	0.1	0.1	0.0	2.43	1.57	1.57	1.49	1.22	1.22
4 sc105	102	0	0	0	145	91	91	56	0.3	0.2	0.2	0.2	2.59	1.63	1.63	1.46	1.06	1.06
5 kb2	0	0	0	0	81	59	59	33	0.1	0.1	0.1	0.1	2.45	1.79	1.79	1.94	1.48	1.48
6 adlittle	78	29	29	16	116	82	82	94	0.2	0.1	0.1	0.1	1.23	0.87	0.87	1.19	0.99	0.99
7 scagr7	146	117	117	51	179	141	141	87	0.3	0.3	0.3	0.2	2.06	1.62	1.62	1.80	1.70	1.70
8 stocfor1	163	76	76	15	194	93	93	31	0.4	0.2	0.2	0.1	6.26	3.00	3.00	5.12	2.12	2.12
9 blend	85	0	0	0	100	98	98	58	0.2	0.2	0.2	0.2	1.72	1.69	1.69	0.92	1.03	1.03
10 sc205	190	0	0	0	320	190	190	129	0.9	0.6	0.6	0.7	2.48	1.47	1.47	1.43	0.96	0.96
11 recipe	68	51	51	12	94	73	73	41	0.1	0.1	0.1	0.1	2.29	1.78	1.78	1.90	1.45	1.45
12 share2b	136	69	69	76	181	107	107	111	0.4	0.2	0.2	0.2	1.63	0.96	0.96	1.46	0.92	0.92
13 vtpbase	248	100	129	87	274	130	151	114	0.7	0.3	0.4	0.3	2.40	1.14	1.32	2.15	0.88	1.12
14 lotfi	183	105	105	68	284	230	230	199	0.6	0.5	0.5	0.6	1.43	1.16	1.16	1.07	0.75	0.75
15 share1b	168	162	162	85	233	240	240	175	0.7	0.7	0.7	0.6	1.33	1.37	1.37	1.11	1.13	1.13
16 boeing2	176	122	97	108	242	188	147	178	0.6	0.5	0.4	0.5	1.36	1.06	0.83	1.22	0.96	0.70
17 scorpion	365	317	306	131	399	382	378	168	2.0	1.7	1.5	0.8	2.38	2.27	2.25	2.45	2.08	1.85
18 bore3d	213	198	198	91	235	221	221	113	0.5	0.5	0.5	0.4	2.08	1.96	1.96	1.26	1.23	1.23
19 scagr25	523	421	418	233	751	565	541	400	5.3	3.7	3.5	3.3	1.88	1.41	1.35	1.61	1.13	1.06
20 sctap1	362	199	180	121	534	350	299	207	2.1	1.4	1.1	0.9	2.58	1.69	1.44	2.39	1.55	1.22
21 capri	461	321	321	344	537	414	417	438	2.3	1.4	1.6	2.2	1.23	0.95	0.95	1.08	0.64	0.73
22 brandy	244	208	208	81	348	326	326	174	1.7	1.5	1.5	1.1	2.00	1.87	1.87	1.51	1.38	1.38
23 israel	298	8	7	7	464	172	204	204	2.4	0.9	1.0	1.0	2.27	0.84	1.00	2.32	0.89	0.99
24 finnis	670	246	255	229	966	461	489	517	5.7	2.4	2.5	3.0	1.87	0.89	0.95	1.93	0.79	0.85
25 gfrdpnc	910	846	846	293	1156	1103	1103	525	6.3	6.2	6.2	3.1	2.20	2.10	2.10	2.00	1.97	1.97
26 scsd1	99	99	99	38	181	181	181	161	0.5	0.5	0.5	0.5	1.12	1.12	1.12	0.90	0.90	0.90
27 etamacro	614	511	489	423	933	764	722	718	4.7	3.6	3.4	3.7	1.30	1.06	1.01	1.28	0.99	0.93
28 agg	585	89	77	54	640	136	116	99	3.9	0.7	0.6	0.6	6.46	1.37	1.17	6.76	1.24	1.10
29 bandm	426	426	426	135	594	594	594	283	4.2	4.2	4.2	1.9	2.10	2.10	2.10	2.18	2.18	2.18
30 e226	290	91	97	34	449	383	360	361	1.9	2.1	1.8	1.9	1.24	1.06	1.00	1.03	1.11	0.97
31 scfxm1	416	258	258	174	520	415	415	324	2.2	1.7	1.7	1.5	1.60	1.28	1.28	1.44	1.13	1.13
32 grow7	0	0	0	0	309	309	309	275	2.2	2.2	2.2	2.5	1.12	1.12	1.12	0.90	0.90	0.90
33 standata	440	179	179	49	522	223	223	153	1.8	0.8	0.8	0.7	3.41	1.46	1.46	2.54	1.11	1.11
34 scrs8	857	879	860	339	1117	1148	1168	561	7.3	7.3	7.7	4.6	1.99	2.05	2.08	1.60	1.60	1.69
35 beaconfd	161	128	128	0	189	163	163	31	0.4	0.4	0.4	0.1	6.10	5.26	5.26	3.58	3.25	3.25
36 boeing1	572	334	341	226	997	710	673	563	4.5	3.4	3.1	3.2	1.77	1.26	1.20	1.38	1.06	0.95
37 shell	619	615	615	260	774	793	793	494	3.8	3.8	3.8	3.0	1.57	1.61	1.61	1.26	1.29	1.29
38 standmps	585	316	313	209	698	455	443	317	2.7	1.8	1.8	1.6	2.20	1.44	1.40	1.70	1.10	1.10
39 stair	492	386	377	227	643	505	568	417	9.2	6.1	7.3	6.6	1.54	1.21	1.36	1.39	0.92	1.11
40 degen2	674	673	659	547	1125	1107	1041	895	11.9	12.7	11.4	9.9	1.26	1.24	1.16	1.19	1.28	1.14
41 agg2	642	72	72	34	769	215	215	125	4.6	1.0	1.0	0.7	6.15	1.72	1.72	6.72	1.53	1.53
42 agg3	614	71	71	36	732	241	241	134	4.7	1.3	1.3	0.7	5.46	1.80	1.80	6.65	1.77	1.77
43 scsd6	221	221	221	114	589	589	589	427	1.7	1.7	1.7	1.4	1.38	1.38	1.38	1.23	1.23	1.23
44 ship04s	413	348	348	182	537	484	484	276	2.0	1.7	1.7	1.2	1.95	1.75	1.75	1.68	1.45	1.45
45 seba	659	659	659	138	853	853	853	234	5.0	5.0	5.0	1.4	3.65	3.65	3.65	3.56	3.56	3.56
46 tuff	377	384	384	366	474	428	428	427	1.9	2.0	2.0	2.4	1.11	1.00	1.00	0.83	0.83	0.83
47 forplan	326	327	327	166	465	472	472	336	1.5	1.6	1.6	1.3	1.38	1.40	1.40	1.14	1.21	1.21
48 bnl1	2241	2147	2147	2401	2511	2384	2384	2642	33.0	30.4	30.4	36.6	0.95	0.90	0.90	0.90	0.83	0.83
49 pilot4	1063	851	729	322	1744	1549	1334	1015	26.5	22.4	18.8	14.6	1.72	1.53	1.31	1.82	1.54	1.29
50 scfxm2	858	602	602	400	1116	884	884	669	9.3	6.3	6.3	5.3	1.67	1.32	1.32	1.74	1.17	1.17

TABLE V

Comparison of Initial Bases:

Artificial (A), Feasible Slack (FS), Slack (S), CPLEX (C)

	Phase I Iterations				Total Iterations				Run Times (seconds)				Total Iterations Ratios			Run Time Ratios		
PROBLEM	A	FS	S	C	A	FS	S	C	A	FS	S	C	A/C	FS/C	S/C	A/C	FS/C	S/C
51 grow15	0	0	0	0	849	849	849	621	11.8	11.8	11.8	11.3	1.37	1.37	1.37	1.04	1.04	1.04
52 perold	3614	3298	3428	1370	4629	4416	4679	2457	100.1	94.3	102.9	52.3	1.88	1.80	1.90	1.91	1.80	1.97
53 fffff800	1106	1058	917	541	1290	1220	1064	683	7.4	7.1	5.9	4.5	1.89	1.79	1.56	1.63	1.58	1.30
54 ship04l	383	360	360	223	565	525	525	353	2.2	2.1	2.1	1.6	1.60	1.49	1.49	1.32	1.30	1.30
55 sctap2	1356	679	611	252	2328	1655	1505	548	29.0	20.1	18.2	4.7	4.25	3.02	2.75	6.24	4.31	3.92
56 ganges	1591	1579	1579	358	1714	1727	1727	605	21.1	21.7	21.7	9.6	2.83	2.85	2.85	2.21	2.27	2.27
57 ship08s	772	697	697	202	1073	955	955	429	7.5	6.7	6.7	3.6	2.50	2.23	2.23	2.07	1.85	1.85
58 sierra	2175	577	577	206	3068	841	841	455	36.3	7.2	7.2	4.4	6.74	1.85	1.85	8.20	1.62	1.62
59 scfxm3	1240	906	906	596	1681	1280	1280	1050	18.8	12.3	12.3	11.6	1.60	1.22	1.22	1.62	1.06	1.06
60 ship12s	1108	1102	1102	316	1346	1287	1287	528	12.6	11.5	11.5	6.3	2.55	2.44	2.44	1.99	1.82	1.82
61 grow22	0	0	0	0	1224	1224	1224	892	23.5	23.5	23.5	21.8	1.37	1.37	1.37	1.08	1.08	1.08
62 stocfor2	2657	1503	1503	411	3508	2182	2182	1061	78.6	46.0	46.0	26.5	3.31	2.06	2.06	2.97	1.74	1.74
63 scsd8	1039	1039	1039	468	2023	2023	2023	1320	12.0	12.0	12.0	8.2	1.53	1.53	1.53	1.47	1.47	1.47
64 sctap3	1703	951	869	337	2896	2273	2221	786	46.5	40.0	39.1	9.2	3.68	2.89	2.83	5.04	4.33	4.24
65 pilotwe	1618	1626	1626	675	3533	3737	3737	2652	75.5	83.4	83.4	76.2	1.33	1.41	1.41	0.99	1.09	1.09
66 maros	1215	742	742	760	1912	1367	1367	1476	22.3	15.7	15.7	18.8	1.30	0.93	0.93	1.19	0.84	0.84
67 fit1p	837	837	837	450	1249	1249	1249	809	16.8	16.8	16.8	11.2	1.54	1.54	1.54	1.49	1.49	1.49
68 25fv47	1760	1414	1377	979	3734	3438	3439	2679	94.1	76.3	77.6	65.1	1.39	1.28	1.28	1.44	1.17	1.19
69 czprob	1421	1245	1245	547	2321	2083	2083	1178	17.8	16.6	16.6	10.8	1.97	1.77	1.77	1.65	1.53	1.53
70 ship08l	771	732	732	326	1387	1283	1283	804	10.3	9.6	9.6	6.6	1.73	1.60	1.60	1.56	1.46	1.46
71 pilotnov	2864	2701	2614	2262	3161	3228	3237	2767	69.5	74.8	75.8	71.2	1.14	1.17	1.17	0.98	1.05	1.06
72 nesm	1777	1951	1951	961	5029	5307	5307	4094	36.6	38.2	38.2	31.0	1.23	1.30	1.30	1.18	1.23	1.23
73 fit1d	0	0	0	0	1161	989	989	947	4.7	4.0	4.0	3.9	1.23	1.04	1.04	1.21	1.05	1.05
74 bnl2	4640	4051	4051	2858	6819	5564	5564	4422	250.5	186.0	186.0	148.2	1.54	1.26	1.26	1.69	1.25	1.25
75 pilotja	4583	3259	3543	2011	7065	5459	6004	4365	207.1	157.7	182.0	139.0	1.62	1.25	1.38	1.49	1.13	1.31
76 ship12l	1220	1147	1147	579	1648	1502	1502	969	18.3	15.8	15.8	12.5	1.70	1.55	1.55	1.46	1.26	1.26
77 cycle	0	0	0	0	2416	2034	2034	1207	38.6	32.2	32.2	27.1	2.00	1.69	1.69	1.42	1.19	1.19
78 80bau3b	4056	2195	2227	2227	11107	10173	10635	10635	173.8	163.0	166.2	166.2	1.04	0.96	1.00	1.05	0.98	1.00
79 degen3	3188	2964	3107	2873	5471	4860	5043	4633	254.7	202.0	204.5	201.4	1.18	1.05	1.09	1.26	1.00	1.02
80 truss	3482	3482	3482	2205	11511	11511	11511	10622	206.8	206.8	206.8	192.4	1.08	1.08	1.08	1.07	1.07	1.07
81 greenbea	3688	3473	3473	2966	7582	8115	8115	7973	282.8	320.3	320.3	341.5	0.95	1.02	1.02	0.83	0.94	0.94
82 greenbeb	3695	3661	3661	3171	6717	7037	7037	6121	250.5	268.5	268.5	241.8	1.10	1.15	1.15	1.04	1.11	1.11
83 d2q06c	4778	3770	2944	1427	10922	10258	9832	9649	684.4	668.6	655.8	684.6	1.13	1.06	1.02	1.00	0.98	0.96
84 woodw	1099	1026	1113	369	2791	2160	2440	1701	38.1	28.6	31.3	25.5	1.64	1.27	1.43	1.50	1.12	1.23
85 pilots	5495	4552	4530	4034	8599	8039	7698	7402	822.7	833.1	757.8	768.3	1.16	1.09	1.04	1.07	1.08	0.99
86 fit2p	5271	5271	5271	3076	12585	12585	12585	10030	796.0	796.0	796.0	690.5	1.25	1.25	1.25	1.15	1.15	1.15
87 stocfor3	21140	12220	12220	5015	28698	18443	18443	10813	4288.5	2762.6	2762.6	1817.5	2.65	1.71	1.71	2.36	1.52	1.52
88 wood1p	211	211	211	222	459	459	459	749	8.7	8.7	8.7	14.0	0.61	0.61	0.61	0.62	0.62	0.62
89 pilot87	5213	4211	4633	4053	10835	10296	9995	9696	2179.0	2176.9	2105.1	2087.2	1.12	1.06	1.03	1.04	1.04	1.01
90 fit2d	2	0	0	0	12593	12041	12041	11726	131.1	125.0	125.1	120.0	1.07	1.03	1.03	1.09	1.04	1.04
TOTALS	122926	94758	94384	58249	226982	197485	197434	157966	11572	9752	9627	8276						
AVERAGES													2.09	1.55	1.54	1.90	1.36	1.35

Acknowledgement

The author would like to thank Bob Fourer for teaching him how to think about the simplex method and Tom Baker for motivating the development of the CPLEX code. Thanks are also due to John Gregory for a number of helpful comments on the original version of this manuscript.

References

[1] Adler, I., N. Karmarkar, M. G. C. Resende, and G. Veiga (1989), An implementation of Karmarkar's algorithm for linear programming, *Mathematical Programming* **44** No. 3, 297–336.

[2] Chvátal, Vašek (1983), *Linear Programming*, W. H. Freeman, New York.

[3] Dantzig, G. B. (1951), Maximization of a linear function of variables subject to linear inequalities, in : *Activity Analysis of Production and Allocation*, R.C. Koopmans (ed.), Wiley, New York, 1951, 339–347.

[4] Golub, G. H. and C. F. Van Loan (1983), *Matrix Computations*, The Johns Hopkins University Press, Baltimore, Maryland.

[5] Optimization Subroutine Library: Guide and Reference (1990), IBM Corporation, April 1990.

[6] Karmarkar, N. (1984), A new polynomial time algorithm for linear programming, *Combinatorica* **4**, 375–395.

[7] Lustig, I. (1987), An analysis of an available set of linear programming test problems, Technical Report SOL 87-11, Systems Optimization Lab, Stanford University, Stanford, California.

[8] Lustig, I. (1987), An analysis of an available set of linear programming test problems, *Comput. Opns Res.* **16** 173–184.

[9] Lustig, I. J., R. E. Marsten, and D. F. Shanno (1990), On implementing Mehrotra's predictor-corrector interior-point method for linear programming, Technical Report SOR 90-03, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.

[10] Markowitz, H. M. (1957), The elimination form of the inverse and its application to linear programming, *Management Science* **3** 255–269.

[11] Murtagh, Bruce A. and Michael Saunders (1987), Minos 5.1 User's Guide, Technical Report SOL 83-20R, Systems Optimization Lab, Stanford University, Stanford, California.

[12] Suhl, Uwe H. and Leena M. Suhl (1990), Computing sparse LU factorizations for large-scale linear programming bases, *ORSA Journal on Computing* **2** 325–335.